Assignment 2
DCS3101

Kent Odde

October 1, 2020

Contents

Abstract

Q1

There are several ways of visualizing an algorithm, i have chosen to include a visual representation first:

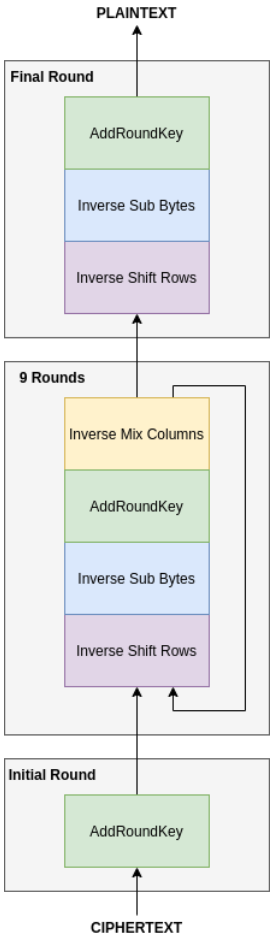

Figure 1: Algorithm for decrypting AES

However, as code is perhaps the most precise way to describe an algorithm, and I already happen to have implemented AES encryption in C++, i decided to also include the decrypt function from that implementation:

```
std::string AES::decrypt128BitMessage(std::string cipherText,
    std::string keyString)
{
   auto key = generateGridFromHexString(keyString);
   auto grid = generateGridFromHexString(cipherText);
   auto expandedKey = generateExpandedKey(key);

   //Initial Round:
   addRoundKey(grid, expandedKey.at(expandedKey.size() - 1));

   //9 Rounds
   for(int i = expandedKey.size() - 2; i > 0; i--)
   {
      invShiftGrid(grid);
      invSubstitute(grid);
      addRoundKey(grid, expandedKey.at(i));
      invMixColumns(grid);
   }

   //Final Round
   invShiftGrid(grid);
   invSubstitute(grid);
   addRoundKey(grid, expandedKey.at(0));

   return gridToHexString(grid);
}
```

Q2

Q3

ECB, CBC, OFB etc. are modes of operation i stream ciphers or when it comes to encrypting data that is longer than the block size given within a certain block cipher algorithm. For instance in AES, a block is 128 bits. If you however want to encrypt more data than this, we use for instance CBC in order to provide additional security across the blocks. The principle is to treat the blocks as a stream.

In this task we have been given these parameters for demonstrating the different modes of operation:

- Plaintext: *DCS-3101*
- IV: *NO*
- Key: *EU*
- Block size: *16 bits*
- Encryption: *XOR*

The block size of sixteen bits means that each block will have two characters, and we will end up with four blocks.

Converted to binary, the plaintext string looks like this:

01000100 01000011 01010011 00101101
00110011 00110001 00110000 00110001

The block size of sixteen bits means that each block will have two characters, and we will end up with four blocks.

DC	:	01000100 01000011
S-	:	01010011 00101101
31	:	00110011 00110001
01	:	00110000 00110001

The key and IV converted to binary looks like this:

IV	:	01001110 01001111
KEY	:	01000101 01010101

1. ECB

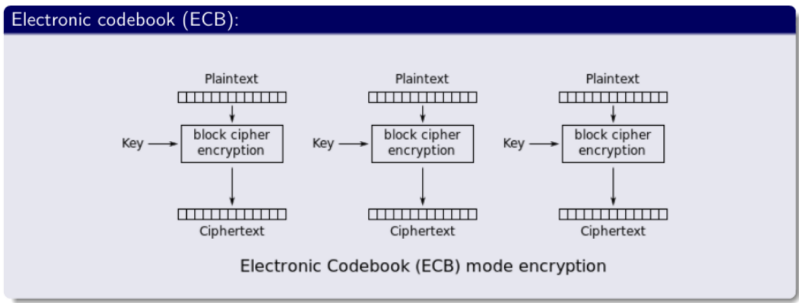

Figure 2: ECB encryption diagram from lecture

ECB is the simplest of the three modes of operation discussed in this assignment, and provide no addition security. The principle is to divide the text into blocks, encrypt them with whatever way we like, and concatenate the ciphertexts produced. This means that if you break one of the blocks, you will be able to break them all. However, an error will not propogate through the blocks.

In practice it will look like this:

```
01000100 01000011   01010011 00101101   00110011 00110001   00110000 00110001
                              XOR
01000101 01010101   01000101 01010101   01000101 01010101   01000101 01010101
                              =
10001011000010110011110000111011001100100011101010110010 0
```

2. CBC

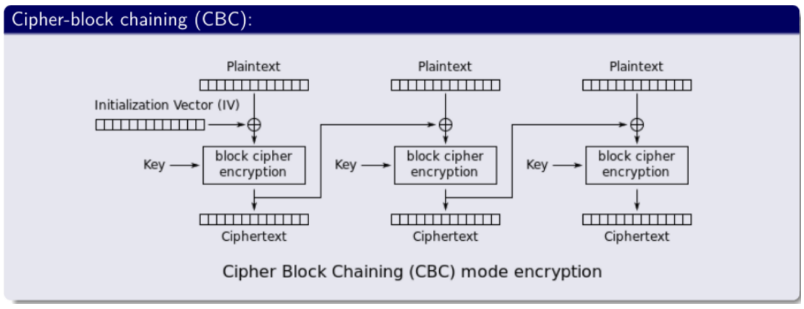

Figure 3: CBC encryption diagram from lecture

3. OFB

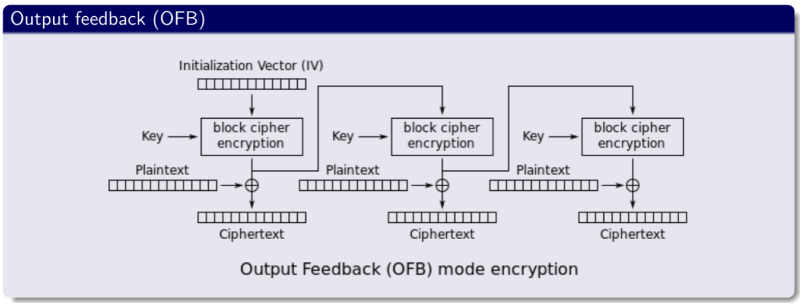

Figure 4: OFB encryption diagram from lecture

4. OFB - Decryption

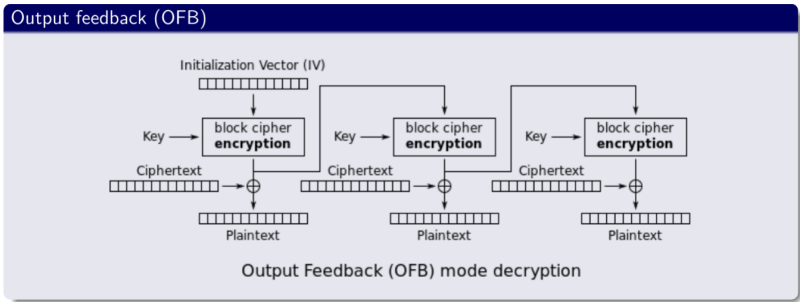

Figure 5: OFB decryption diagram from lecture

Q4

Appendices